REMARKS

Claims 1, 5, 8-10, 35-40, 42, and 53 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent No. 5,870,611 (London Shrader) in view of U.S. Patent No. 5,960,189 (Yinger). Reconsideration is respectfully requested in view of the above amendments and the following remarks.

Interview Summary

On July 17, Applicant's undersigned representative conducted a telephonic interview with Examiner Dao. During that interview, the London Shrader reference and the pending claims were discussed.

Applicant's undersigned representative wishes to thank Examiner Dao for conducting the telephonic interview and for his continued consideration of the present application.

Prior Art Rejections

The present application discloses an application manager and methods performed by the application manager.

> Generally, the application manager controls operations related to the installation of data objects on a computer system. In particular, the application manager manages the following operations: install, downsize, reinstall, and uninstall. The application manager *relies upon the data objects to physically implement each of the installation operations* but directs the data objects with respect to which operations should be performed and when the operations should be performed. (Application, p. 4, ll. 14-20) (emphasis added).

> [A]n application programming interface (API) is provided which allows data objects, such as software applications to interface with the application manager. (Application, p. 4, ll. 11-13).

> The application manager operates on a notify-and-commit procedure with respect to the four installation operations. Accordingly, *prior to executing one of the installation operations, a data object must notify the application manager of the impending operation*. Furthermore,

> *after completing the installation operation, the application finalizes the operation by committing it with the application manager.* (Application, p. 4, ln. 23 – p. 5, ln. 5) (emphasis added).

In connection with Figure 8, the application discloses an exemplary process for controlling an installation operation.

> Figure 8 provides a flowchart of a generalized process for implementing the four application installation operations: install, downsize, reinstall, and uninstall. The process of Figure 8 applies generally to each of these installation operations, although slight variations, which are described below, may apply to the individual operations.
>
> Prior to implementing an installation operation, *an application* needs to set certain properties that are associated with an instance of the IapplicationEntry object. Accordingly, as shown at step 810, the *application manager receives a call* to the SetProperty() method *from the application*. The SetProperty() method initializes application properties associated with the IapplicationEntry object.
>
> After values have been established for the necessary properties, *an application can initialize* the installation operation. The initialize procedure is equivalent to notifying the application manager that the application is undertaking to perform a particular installation operation. Accordingly, at step 812, *a call* to one of the initialize methods, InitializeInstall(), InitializeReinstall(), InitializeUninstall(), or InitializeDownsize() is *received by the application manager*. The initialize call notifies the application manager of the impending application installation.
>
> If *the application* did *not* successfully *complete* the installation operation(step 814), at step 816, a call to the Abort() method of the IapplicationEntry object is *received from the application*.
>
> If, however, *the application* successfully *completed* the installation operation, at step 818, *a call* to one of the finalize methods FinalizeInstall(), FinalizeReinstall(), FinalizeUninstall(), or FinalizeDownsize() is received *by the application manager*. The finalize method operates to commit to the application manager that the particular installation

operation has been completed. (Application, p.48, ln. 15 – p. 49, ln. 23).

Claim 1 of the application is directed to a method of communicating with an application in a system for managing application installation operations. The recited method comprises:

> **at an application manager** adapted to control installation operations on a computing system,
> **communicating to an application** an instruction to perform a downsize operation, said instruction **identifying an amount of space to be freed by the application**;
> **at the application manager receiving from the application a call to set a property related to performing an application installation operation by the application**, wherein the application installation operation is a downsize operation, further wherein the downsize operation comprises one of removing non-essential data and removing data that can be recreated from another source;
> **at the application manager receiving from the application a call to initialize the application installation operation, said call** to initialize the application installation operation **notifying the application manager of an impending application installation operation to be performed by the application**;
> **at the application manager receiving from the application a call to finalize the application installation operation, said call** to finalize the application installation operation **communicating to the application manager the application installation operation has been completed by the application**; and
> if the application installation operation is not executed successfully by the application, **at the application manager receiving from the application a call to abort the application installation operation**.

In order for a reference to anticipate, or a set of references to render claim 1 obvious, the references must disclose or suggest the entire process, including each of the recited features. Applicant's undersigned representative respectfully submits that the cited references do not teach or suggest these features.

London Shrader discloses systems and methods for defining and constructing a proposed plan object for installing software across a network. (London Shrader., col. 1, ll. 18-

22). The disclosed system alleges to reduce the network installation planning process into a series of discrete objects and provides a graphical interface by which administrators can select applications for installation operations on a set of workstations across a LAN. (London Shrader, col. 2, ll. 7-13). London Shrader discloses a sample user interface in connection with Figure 17. The administrator uses the interface to generate files that are used to perform physical installation. (London Shrader, col. 2, ll. 13-15). The files may specify, amongst other things, installation commands (installation, configuration, removal, and reinstallation) that are to be performed in connection with applications. The files that are generated by the adminstrator's user interface application are then used *by an installation engine* to actually perform the installation operation across the network. (London Shrader, col. 5, ll. 56-57).

Thus, London Shrader discloses a graphical user interface by which an administrator can identify applications for installation and generate files for use in performing the installation operation, and then have an *installation engine* use the files to perform the installation operation. In contrast, claim 1 recites a method wherein the *applications themselves* perform the installation operations and *communicate with an application manager* regarding the operations that the applications will and/or have already performed.

Claim 1 recites "**at an application manager** . . . **communicating to an application** an instruction to perform a downsize operation, said instruction **identifying an amount of space to be freed by the application.**" The concept of communicating at an application manager *to an application* an instruction to perform a downsize operation is completely absent from the system taught by London Shrader. In the system disclosed by London Shrader, an *installation engine* performs the installation operations, not the applications themselves as recited in claim 1. (London Shrader, col. 5, ll. 56-57). In the system taught by London Shrader, there is simply no reason to communicate an instruction identifying the amount of space to be freed by the application from an application manager to an application.

The office action alleges that London Shrader teaches receiving from the application a call to set a property related to performing an application installation operation at column 7, lines 21 through 30. (Office Action, page 3). It is true that the cited portion of London Shrader discloses setting attributes used to denote the type of processing (installation, configuration, removal, or reinstallation) to be performed on an object. (London Shrader et al., col. 7, ll. 28-30). But in contrast to claim 1, London Shrader, including the referenced

section does *not* teach "**at an application manager . . .receiving from an application a call to set a property related to performing an application installation operation by the application**." The concept of receiving *from an application* a call related to an application installation operation that is performed *by the application* is completely absent from the system taught by London Shrader. In the system disclosed by London Shrader, an *installation engine* performs the installation operations, not the applications themselves as recited in claim 1. (London Shrader, col. 5, ll. 56-57). Accordingly, in London Shrader there are no calls from an application to an application manager relating to installation operations performed by the application. Indeed, by teaching a system wherein an installation engine, and not the application, performs installation operations, London Shrader actually teaches away from a method where such a step is performed.

The office action alleges that London Shrader teaches (col. 7, ll. 21-3) receiving from the application a call to initialize an application installation operation (Office Action, page 4). As acknowledged above, it is true that the cited portion of London Shrader discloses setting attributes used to denote the type of processing (installation, configuration, removal, or reinstallation) to be performed on an object (London Shrader et al., col. 7, ll. 28-30). But in contrast to the language of claim, London Shrader, including the referenced section does *not* teach "**at the application manager** receiving **from the application** a call to initialize the application installation operation, said call . . . notifying the application manager of an impending application installation operation **to be performed by the application**." The concept of receiving *from an application* a call notifying the application manager of an impending installation operation to be performed *by the application* is completely absent from the system taught by London Shrader. In the system disclosed by London Shrader, *an installation engine* performs the installation operations, not the applications themselves. (London Shrader, col. 5, ll. 56-57). In London Shrader, there is no need for calls from an application notifying an application manager of impending installation operations performed by the application. In fact, by teaching a system wherein an installation engine, and not the application, performs installation operations, London Shrader actually teaches away from a method such as recited in the claim.

The office action alleges that London Shrader teaches receiving from the application a call to finalize the application installation operation (Office Action, page 5). Admittedly, the

cited portion of London Shrader discloses setting attributes used to denote the type of processing (installation, configuration, removal, or reinstallation) to be performed on an object (London Shrader et al., col. 7, ll. 28-30). But in contrast to the language of the claim, London Shrader, including the referenced section does *not* teach "**at the application manager** receiving **from the application** a . . . call to finalize the application installation operation communicating to the application manager the application installation operation **has been completed by the application**." The concept of receiving *from an application* a call communicating *to the application manager* that the application installation operation has been completed *by the application* is completely absent from the system taught by London Shrader. Indeed, in the system disclosed by London Shrader, an installation engine performs the installation operations, not the applications themselves. (London Shrader, col. 5, ll. 56-57). Accordingly, in London Shrader, there is no need for applications communicating calls from an application informing the application manager of installation operations that have been performed by the application. Indeed, by teaching a system wherein an installation engine performs installation operations, London Shrader actually teaches away from the recited claim language.

The second reference relied upon in the office action, Yinger, discloses a computer application for installing new applications and updating existing applications on a computer node within a computer network environment. (Yinger, col. 1, ll. 49-52). In the system disclosed by Yinger, in response to a client computer receiving a request from a user to run an application, the client computer determines whether the application exists on the client computer. (Yinger, col. 1, ll. 59-61). In response to the application existing on the client computer, the client computer runs the application. (Yinger, col. 1, ll. 61-62). Otherwise, if the desired application is available on a server computer, *the server computer* automatically *installs* the application on the client computer, which then automatically executes the application after installation. (Yinger, col. 1, ll. 62-67). The section of Yinger that is referenced in the office action refers to communications *internal* to an application that loads application modules into memory for execution. (Yinger, Col. 12, ll. 7-9). Thus, Yinger teaches a server computer that automatically installs applications. Yinger does not teach systems and methods wherein an *application manager receives communications from applications* regarding the installation operations that the *applications themselves* are making.

Therefore, because London Shrader and Yinger fail to teach or even suggest the recited features, claim 1 is neither anticipated nor rendered obvious. For similar reasons, claim 35 is neither anticipated nor rendered obvious. Withdrawal of the prior art rejections is respectfully requested.

CONCLUSION

The undersigned respectfully submits that pending claims are allowable and the application in condition for allowance. A Notice of Allowance is respectfully solicited.

Examiner Dao is invited to call the undersigned in the event a telephone interview will advance prosecution of this application.

Date: July 23, 2007

/John E. McGlynn/
John E. McGlynn
Registration No. 42,863

Woodcock Washburn LLP
Cira Centre
2929 Arch Street, 12th Floor
Philadelphia, PA 19104-2891
Telephone: (215) 568-3100
Facsimile: (215) 568-3439